**Experiment 1.1**

**Student Name:** Gaurav Kumar**UID:** 22MCC20177

**Branch:**MCA CC & Devops**Section/Group:** 22MCD-1-A Group

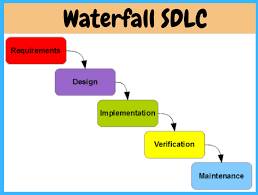
**Semester:**2**Date of Performance:** 13-02-2023

**Subject Name:** Software Testing **Subject Code:** 22CAH-685

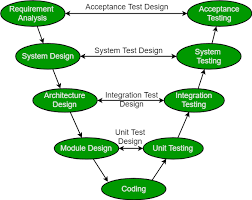
**Aim/Overview of the practical:**

Testing principles through illustrations with respect to different SDLC Models i.e., Waterfall, V-Shaped, RAD and Agile.

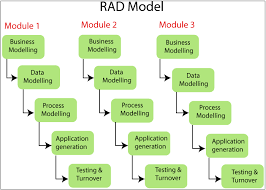
**Theory :**

Seven testing principles with respect to **Waterfall SDLC Models:**

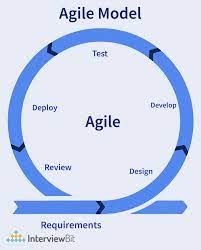
1. **Early Testing:** In Waterfall SDLC, testing is typically conducted in the later stages of the software development process after the requirements, design, and development have been completed. This can lead to delays in detecting defects and may result in higher costs to fix them.
2. **Automation:** In Waterfall SDLC, manual testing is more prevalent, which can be time-consuming and prone to errors. Automated testing is usually limited to specific regression tests, which are run repeatedly in later stages of the development process.
3. **Exhaustive Testing is impossible:** In Waterfall SDLC, testing is usually limited to predefined test cases. It's impossible to test every possible combination of inputs and scenarios in a complex software system, so some defects may go undetected until later stages of the development process.
4. **Defect Clustering:** In Waterfall SDLC, defects are usually detected and fixed in later stages, which can lead to an increased number of defects and failures. This can result in a defect clustering effect, where a small number of defects contribute to a large number of failures.
5. **Pesticide Paradox:** In Waterfall SDLC, the same set of tests is usually repeated in each testing phase. This can lead to the identification of fewer defects as the same tests are run repeatedly.
6. **Testing is context-dependent:** In Waterfall SDLC, testing is usually conducted in a controlled environment, where the system is tested in isolation from other systems. This can lead to a lack of consideration for how the system will interact with other systems and can result in defects that are only discovered in the later stages of the development process.
7. **Continuous Improvement:** In Waterfall SDLC, testing is typically conducted in a fixed and rigid manner, with limited feedback and opportunities for improvement throughout the development process. This can result in missed opportunities to improve the testing process and increase the quality of the software being developed.

****Seven testing principles with respect to **V-Shaped SDLC Models:**

* 1. **Early Testing:** In V-shaped SDLC, testing is conducted at each stage of the development process in parallel with the corresponding development activities. This ensures that defects are detected early and can be fixed before they are propagated to later stages of the development process.
  2. **Automation:** In V-shaped SDLC, automated testing is used to streamline the testing process and increase the efficiency and effectiveness of testing. Automated tests are run at each stage of the development process, ensuring that any defects are detected early and can be fixed before they impact the quality of the software being developed.
  3. **Exhaustive Testing is impossible:** In V-shaped SDLC, testing is usually limited to predefined test cases. It's impossible to test every possible combination of inputs and scenarios in a complex software system, so some defects may go undetected until later stages of the development process.
  4. **Defect Clustering:** In V-shaped SDLC, defects are detected and fixed at each stage of the development process, ensuring that they do not propagate to later stages and result in an increased number of defects and failures.
  5. **Pesticide Paradox:** In V-shaped SDLC, the same set of tests is usually repeated at each stage of the development process. This can lead to the identification of fewer defects as the same tests are run repeatedly.
  6. **Testing is context-dependent:** In V-shaped SDLC, testing is conducted in a controlled environment, where the system is tested in isolation from other systems. This can lead to a lack of consideration for how the system will interact with other systems and can result in defects that are only discovered in the later stages of the development process.
  7. **Continuous Improvement:** In V-shaped SDLC, testing is typically conducted in a fixed and rigid manner, with limited feedback and opportunities for improvement throughout the development process. This can result in missed opportunities to improve the testing process and increase the quality of the software being developed. However, V-shaped SDLC does allow for continuous improvement through feedback obtained from each stage of the development process. This feedback can be used to improve the testing process and increase the quality of the software being developed.

Seven testing principles with respect to **RAD SDLC Models:**

1. **Early Testing:** In RAD SDLC, testing is conducted in parallel with the development process, ensuring that defects are detected early and can be fixed before they impact the quality of the software being developed.
2. **Automation:** In RAD SDLC, automated testing is used to streamline the testing process and increase the efficiency and effectiveness of testing. Automated tests are run continuously throughout the development process, ensuring that any defects are detected early and can be fixed before they impact the quality of the software being developed.
3. **Exhaustive Testing is impossible:** In RAD SDLC, testing is usually limited to predefined test cases. It's impossible to test every possible combination of inputs and scenarios in a complex software system, so some defects may go undetected until later stages of the development process.
4. **Defect Clustering:** In RAD SDLC, defects are detected and fixed continuously throughout the development process, ensuring that they do not propagate to later stages and result in an increased number of defects and failures.
5. **Pesticide Paradox:** In RAD SDLC, the same set of tests is usually repeated continuously throughout the development process. This can lead to the identification of fewer defects as the same tests are run repeatedly.
6. **Testing is context-dependent:** In RAD SDLC, testing is usually conducted in a controlled environment, where the system is tested in isolation from other systems. This can lead to a lack of consideration for how the system will interact with other systems and can result in defects that are only discovered in the later stages of the development process.
7. **Continuous Improvement:** In RAD SDLC, the development process is iterative and feedback-driven, which allows for continuous improvement of the testing process throughout the development cycle. This feedback can be used to improve the testing process and increase the quality of the software being developed. The RAD SDLC model is focused on delivering a functional system quickly, so there is a greater emphasis on continuous improvement to ensure that the software meets the requirements and is of high quality.

Seven testing principles with respect to **Agile SDLC Models:**

1. **Early Testing:** In Agile SDLC, testing is conducted continuously throughout the development process, ensuring that defects are detected early and can be fixed before they impact the quality of the software being developed.
2. **Automation:** In Agile SDLC, automated testing is used extensively to streamline the testing process and increase the efficiency and effectiveness of testing. Automated tests are run continuously throughout the development process, ensuring that any defects are detected early and can be fixed before they impact the quality of the software being developed.
3. **Exhaustive Testing is impossible:** In Agile SDLC, testing is usually limited to predefined test cases. It's impossible to test every possible combination of inputs and scenarios in a complex software system, so some defects may go undetected until later stages of the development process.
4. **Defect Clustering:** In Agile SDLC, defects are detected and fixed continuously throughout the development process, ensuring that they do not propagate to later stages and result in an increased number of defects and failures.
5. **Pesticide Paradox:** In Agile SDLC, the same set of tests is usually repeated continuously throughout the development process. This can lead to the identification of fewer defects as the same tests are run repeatedly.
6. **Testing is context-dependent:** In Agile SDLC, testing is usually conducted in a controlled environment, where the system is tested in isolation from other systems. However, the Agile approach also emphasizes testing in real-world scenarios, allowing for consideration of how the system will interact with other systems and end-users. This can result in fewer defects and better overall software quality.
7. **Continuous Improvement:** In Agile SDLC, the development process is iterative and feedback-driven, which allows for continuous improvement of the testing process throughout the development cycle. This feedback can be used to improve the testing process and increase the quality of the software being developed. The Agile SDLC model places a strong emphasis on continuous improvement to ensure that the software meets the evolving needs of the customer and is of high quality